

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

Public Reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comment regarding this burden estimates or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave Blank)		2. REPORT DATE August 20, 2003	3. REPORT TYPE AND DATES COVERED Final Report - Feb. 1, 2003 - July 31, 2003
4. TITLE AND SUBTITLE A JDBC Driver Supporting Data Source Integration and Evolution			5. FUNDING NUMBERS DAAD19-02-1-0455
6. AUTHOR(S) Dr. Ramon Lawrence			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) University of Iowa Division of Sponsored Programs 2 Gilmore Hall			8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) U. S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211			10. SPONSORING / MONITORING AGENCY REPORT NUMBER 44366.1-C1-11
11. SUPPLEMENTARY NOTES The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.			
12 a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			12 b. DISTRIBUTION CODE
13. ABSTRACT (Maximum 200 words) The objective of this work is to create a Java DataBase Connectivity (JDBC) driver that allows access to multiple data sources simultaneously. Unlike existing relational database systems and query tools, the JDBC driver will have several important benefits: 1) The driver can be integrated into existing and future Java programs and be used to isolate applications from integration challenges. 2) A semantic query language that allows users and applications to specify queries without referencing exact structures. 3) Ability to handle incomplete, inconsistent, or missing data. 4) Automatic, dynamic integration for fast information fusion of multiple sources. The integration result can be iteratively refined as more information about the data source structure and contents become available. The contribution of this work is two-fold. First, cutting-edge research will be performed on data source fusion and query interfaces for integration systems. Second, an actual integration product, the JDBC driver, will be produced that will allow the Army and others to exploit integration research in a convenient form for ongoing development and testing.			
14. SUBJECT TERMS database integration, information fusion, inconsistent data, incomplete data, JDBC, Java			15. NUMBER OF PAGES 3
			16. PRICE CODE
17. SECURITY CLASSIFICATION OR REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION ON THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)
Prescribed by ANSI Std. Z39-18
298-102

Final Progress Report
Grant # - DAAD19-02-1-0455
A JDBC Driver Supporting Data Source Integration and Evolution

Statement of the Problem Studied

The problem studied is how to rapidly integrate information from multiple data sources. Current approaches perform integration [Halevy01] by building a global view and then mapping queries on the global view to the data sources. Building a global view is still performed using manual techniques [Batini86]. *Thus, integration is costly and time-consuming because building a global view is a bottleneck in the process.* Further, although many integration systems and prototypes have been developed [Goh99,Kirk95,Li98], none have remained as viable, usable products. The reason for this is that they were built using proprietary technology and require expertise out of the realm of most developers and users.

The goal of this research is to demonstrate that practical, rapid information fusion can be achieved by:

- Building an integration architecture using common industrial standards such as Java and Java DataBase Connectivity (JDBC).
- Developing a system for documenting data source contents so that they can be rapidly shared and integrated.
- Defining a high-level query language that allows users to specify the concepts they want without indicating how to retrieve them. The language must hide integration details and be easier to use than SQL.
- Supporting "real data" by handling data inconsistency, incompleteness, and outlying data such that data mining and decision support systems can identify meaningful outlying data that is not filtered out by the integration system.

The research product of this project is a JDBC driver that allows for Java programs to transparently query data sources without specifying structural queries (SQL). Programs and users specify semantic queries to the JDBC driver that translates the high-level queries into SQL queries for the appropriate data sources. This automatic translation process performed by the driver isolates users and applications from the complexity of multiple data source querying and allows the applications to function in the presence of schema evolution of the underlying data sources.

Summary of Important Results

The major product of this research is a JDBC driver (see Figure 1) capable of integrating multiple data sources. The JDBC driver processes high-level user queries and converts them to queries on multiple databases. Information from the multiple databases is fused together and presented to the user. A unique feature of the driver is that it supports data inconsistency. Information that is inconsistent across the databases is highlighted and can be used to determine data deserving further investigation.

This proof of concept implementation demonstrates that it is feasible to integrate databases using a JDBC driver. Using the driver, Java applications can be rapidly developed that extract information from multiple sources. Since the query language does not force the user to reference particular databases, tables, or fields, developing an application that access multiple databases is no more complex than developing an application that accesses a single database. Further, the system supports data source evolution as the mapping process performed inside the driver allows the databases queried to change without affecting user queries.

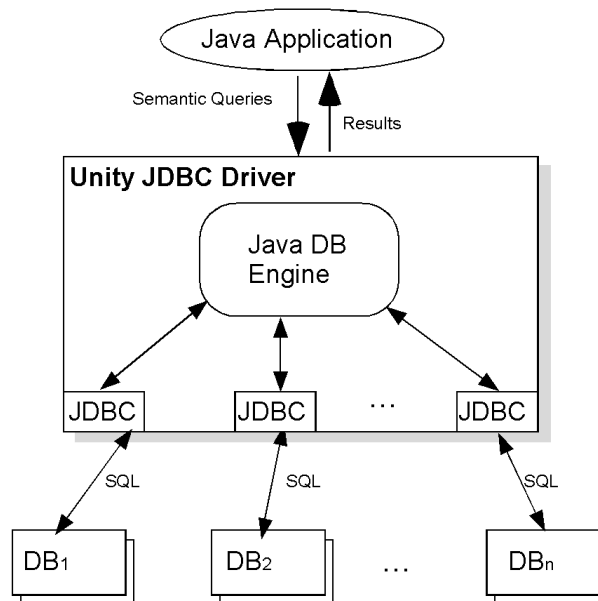


Figure 1. Unity JDBC Driver Architecture

The major important results are:

- A JDBC driver implementation based on the Unity architecture [Lawrence01], called *UnityDriver*, that supports multiple database querying.
- Demonstration of how *UnityDriver* can be used as platform for developing applications for performing data mining and information fusion.
- A high-level query language allowing users to easily query multiple databases.
- A mapping algorithm for converting high-level queries into SQL and integrating results returned.

There are two keys to the success of the integration. First, databases are *annotated* with more information so that the concepts in them can be more rapidly compared. One of the keys to successful integration is assigning meaningful names, so that users can query on familiar names rather than obscure system names. The second key is the ability to automatic insert local and global joins in a user query. A user may request the system: “return all soldiers who have chemical training and are currently stationed in Iraq”. The system would determine where those concepts are in the underlying databases and how to combine joins within and across databases to answer the user query *without the user’s involvement*. The algorithms for automatic join determination are unique to this work and will be the subject of future publications.

Publications and Reports

The JDBC driver implementation can be downloaded at <http://idealab3.cs.uiowa.edu>. Included is documentation on how to use the driver and sample programs. The driver was tested on integration problems. The test programs can be used over the Internet and are available at <http://idealab3.cs.uiowa.edu>.

A description of the JDBC driver implementation will be made available in a University of Iowa technical report and will be submitted for publication in 2004.

Participating Researchers and Supported Students

- Dr. Ramon Lawrence - Principal Investigator
- Terry Mason - Ph.D. student
- Jian Jia - Master's student - completed degree while working on the project

Bibliography

[Batini86] Batini, C., Lenzerini, M. and Navathe, S. (1986) A Comparative Analysis of Methodologies for Database Schema Integration, *ACM Computing Surveys*, 18(4), pages 323-364.

[Goh99] Goh, C., Bresson, S., Madnich, S. and Siegel, M. (1999) Context Interchange: New Features and Formalisms for the Intelligent Integration of Information, *ACM Transactions on Information Systems*, 17(3), 270-293.

[Halevy01] Halevy, A. (2001) Answering queries using views: A survey, *VLDB Journal*, 10(4), pages 270-294.

[Kirk95] Kirk, T., Levy, A., Sagiv, Y. and Srivastava, D. (1995) The Information Manifold, *AAAI Spring Symposium on Information Gathering*.

[Lawrence01] Lawrence, R. and Barker, K. (2002) Using Unity to Semi-Automatically Integrate Relational Schema, Demonstration at *International Conference of Data Engineering (ICDE 2002)*, pages 329-330.

[Li98] Li, C., Yerneni, R., Vassalos, V., Garcia-Molina, H., Papakonstantinou, Y., Ullman, J., and Valiveti, M. (1998) Capability Based Mediation in TSIMMIS, *Proceedings of the ACM SIGMOD Conference*, pages 564-566.